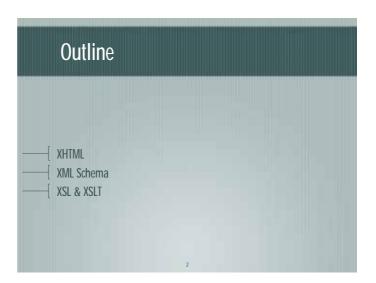


XML Applications

Prof. Andrea Omicini **DEIS**, Ingegneria Due Alma Mater Studiorum, Università di Bologna a Cesena andrea.omicini@unibo.it





HTML vs. XML HTML Presentation oriented No structure, no semantics for data XML Data oriented Allows for structural / semantic representation Can be validated through grammars

XHTML: An XML-based HTML

The idea: use XML rather than SGML to define an HTML equivalent — so, XHML is an XML application — keeping most HTML tags with their original semantics

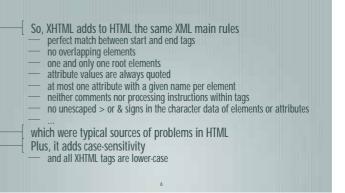
but!

with the properties of well-formedness and validability of XML In fact, most browsers have extended support from HTML to XHTML soon and easily

http://www.w3.org/MarkUp/2004/xhtml-faq Standard W3C

"The Extensible HyperText Markup Language (XHTML[™]) is a family of current and future document types and modules that reproduce, subset, and extend HTML, reformulated in XML" — XHTML 1.0, 1.1, 2.0, Basic, etc.

Main differences



An XHTML Fragment

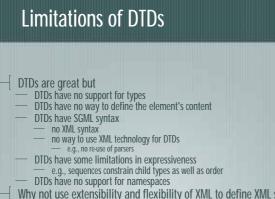
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"> <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
</html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"></html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"></html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"></html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"></html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"></html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"></html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"></http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml" xml:lang="en"></http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http://www.w3.org/1999/xhtml</http:

-

- <div class="body">

</di v>

</body> </html >



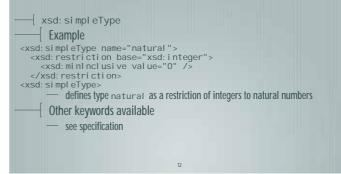
Why not use extensibility and flexibility of XML to define XML syntax? — using XML as a meta-markup language to define a new XML application?

XML Schema

Goals of XML Schemas



Elements of XML Schemas: Simple Type Elements



Elements of XML Schemas: Pre-defined Simple Type Elements

For a type system to be supported, first some pre-defined types should be provided

- string, boolean, float, double, integer
- date binary
- uriReference
- pattern

Then, you can define your own simple types

Elements of XML Schemas: Complex Type Elements

— xsd: compl exType
Example
<pre><xsd: compl="" extype="" name="compl ex"></xsd:></pre>
<pre><xsd: el="" ement="" name="real" type="xsd: float"></xsd:></pre>
<pre><xsd: el="" ement="" name="imaginary" type="xsd: fl oat"> </xsd:></pre>
defines type complex as a pairing of real numbers
Using element declarations
— most of the facets for simple types can be used as attributes for elements
— e.g., mi nI ncl usi ve,

Elements of XML Schemas: Element Declarations

— xsd:element
——[Examples
xsd:element name="point" type="complex"> xsd:element name="goals" type="natural">
Element declaration associates types to elements
— from pre-defined, simple to complex types
Element declarations make a given element admissible within the doc
— again, what is not specified is not allowed
What is missing now are attribute declarations

Elements of XML Schemas: Attribute Declarations

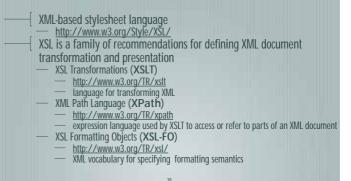
xsd: attri bute
 Example
 xsd: attri bute name="team" type="string">
 xsd: attri bute name="team" type="bool ean" use="required" default="false">
 All attributes are declared as simple types
 Only complex elements can have attributes
 Attribute declarations make a given attribute admissible for an element of a given complex type within the doc

XSL & XSLT

Elements of XML Schemas: Last Few Things

<xsd: ns:="" schema="" xml="" xsd="http://www.w3c.org/2001/XMLSchema"></xsd:>
Associates the XML Schema namespace to the xsd prefix
— Just after the XML Declaration
— since and XML Schema is first of all an XML document
<pre><xsd: compl="" extype="" mixed="true"></xsd:></pre>
Complex Types are allowed to specify Mixed Content
 for mixed-content, narrative-oriented XML documents

XSL: eXtensible Stylesheet Language



XSL Transformations

-{	XSLT is a language for transforming the structure of an XML document Why Transform XML?
	— Two main issues for XML
	— data separation from presentation
	— portability / transmission of information
r	- often, the two together
	In any case, this means that XML documents are typically NOT used in
	the same form they come in
	— thus, the need to transform XML documents
_	Also, DOM and SAX allow for XML transformation
L	— they are similar, and also procedural
	— a more high-level, declarative form should be possible
	- which is where XSLT comes in
	19

An Example: Hello World, XML



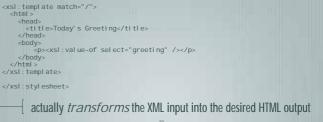


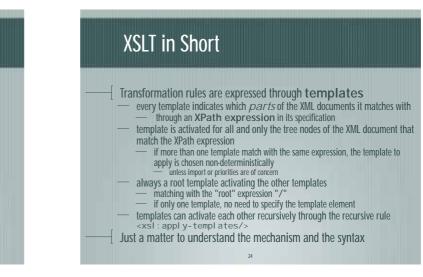
Experiments

A meta-processor for XSLT

Browsers







Another Example of a XSLT sheet



Where to Use XSLT?

different ways to represent the same things

 chunks of knowledge from different sources to be put together from XML to XML
 but also from anything to anything, just using the right parser / writer

through a possibly huge range of different media and scenarios
 XML handles knowledge independently of the presentation
 but then presentation is often needed in the end
 And, the two things together, more often today

Data Conversion scenarios — when there are

Publishing scenarios — typically meant to humans

XSLT is Declarative

XPath	
 Expressions are part of the XSL specification defined as stand-alone component since they are used in other contexts, su as XLink & XPointer Used throughout XSLT to select data from the source and manipulate Syntax defined through <i>production rules</i> like many grammars you already know, maybe The language is complex and articulated better to learn by need, for you Examples chapter//footnote selects all the child node footnote of node chapter which is child of the context node attri bute:: col our selects the col our attribute of the context node 	e it

CSS vs. XSL

CSS advantages

XSL advantages

scopes

What to choose between CSS and XSL? — CSS and XSL overlap to some extent

so they can live together for a while

simple, specific, well supported by all browsers

in the long run, XSL is the obvious front-runner

more powerful, more general, goes far beyond mere presentation

So, even though they overlap a bit, they have different goals and

but simplicity, support and legacy have often won over any other consideration

XML Formatting Objects (XSL-FO)

- XML application to describe the layout of a page / presentation a sort of page-description language à la PostScript, without a programing language
- XSL-FO provides a more sophisticated and flexible visual layout model than $\mbox{HTML} + \mbox{CSS}$
 - like right-to-left and top-to-bottom text, footnotes, margin notes, page numbers in cross-references, etc.
 - more or less generalises over HTML+CSS
 in fact, you may easily find the same property specification as CSS
- 56 elements
 - in the <u>http://www.w3.org/1999/XSL/Format</u> namespace
 rectangular areas with formatting properties

29